3 UTILITY

- There's a **ton** of helpers and utilities in Compass: http://compass-style.org/reference/compass
- We'll be going through a few of our favorites
- Not everything is optimal or current - **always** be aware of the output

3.1 Use With Care

application.sass

```
@import "compass"

%group,
.group
  +clearfix
```

application.css

```
.group {
  overflow: hidden;
  *zoom: 1;
}
```

3.1 Use With Care

application.sass


```
@import "compass"

%group,
.group
  +clearfix
```

application.sass


```
%group,
.group
  &:after
    content: ''
    display: table
    clear: both
```

3.1 Use With Care

`image-url` and `font-url` are available post-configuration:

application.sass

```
@import "compass"

body
  background: image-url('bg-body.png') repeat
```

application.css

```
body {
  background: url('/images/bg-body.png') repeat;
}
```


`scale-lightness` shortcuts our `scale_color` use:

application.sass

```
.content
  color: scale_color(#eee, $lightness: 7%)
```

application.sass

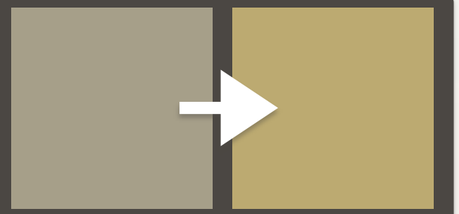
```
@import "compass"

.content
  color: scale-lightness(#eee, 7%)
```

use a negative value to darken

`scale-saturation` also shortcuts `scale_color`:

application.sass

```
.content
  color: scale_color(#a99f88, $saturation: 30%)
```



application.sass

```
@import "compass"

.content
  color: scale-saturation(#a99f88, 30%)
```

Assembly Tip

Color obsessed? Additional Compass color functions include `shade` and `tint`

Opposite Position

Returns the opposite side (or pair):

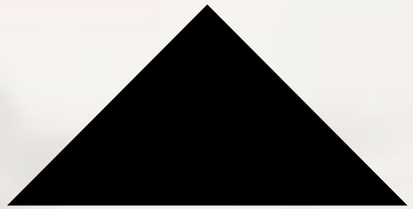
application.sass

```
@import "compass"

opposite-position(top) // bottom
opposite-position(left) // right
opposite-position(right bottom) // left top
```


Using `opposite-position` - CSS shapes:

application.sass

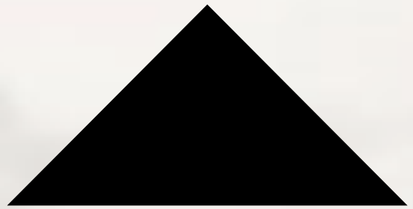
```
.caret
  border: 100px solid transparent
  border-bottom: 100px solid #000
  border-top: 0
  height: 0
  width: 0
```



Using `opposite-position` - CSS shapes:

application.sass

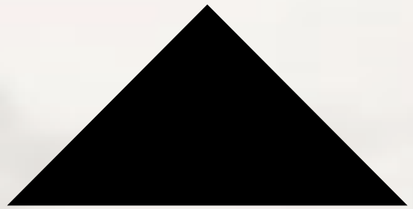
```
=caret($point)
  border: 100px solid transparent
  border-bottom: 100px solid #000
  border-top: 0
  height: 0
  width: 0

.caret
  +caret(top)
```



Using `opposite-position` - CSS shapes:

application.sass

```
=caret($point)
  $opposite: opposite-position($point)
  border: 100px solid transparent
  border-bottom: 100px solid #000
  border-top: 0
  height: 0
  width: 0

.caret
  +caret(top)
```



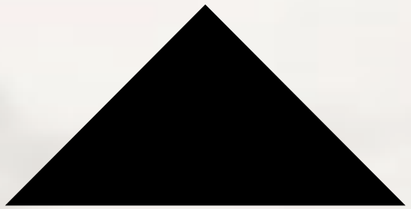
Using `opposite-position` - CSS shapes:

application.sass

```
=caret($point)
  $opposite: opposite-position($point)
  border: 100px solid transparent
  border-#{$opposite}: 100px solid #000
  border-#{$point}: 0
  height: 0
  width: 0

.caret
  +caret(top)
```

top

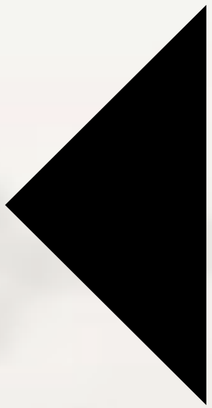
bottom



Using `opposite-position` - CSS shapes:

application.sass

```
=caret($point)
  $opposite: opposite-position($point)
  border: 100px solid transparent
  border-#{$opposite}: 100px solid #000
  border-#{$point}: 0
  height: 0
  width: 0

.caret
  +caret(left)
```



Assembly Tip

Responsive graphs?
Compass adds math functions like `pi`, `sin`, `cos`, `tan`, `e`, `log`, `sqrt`, and `pow`

Contrast Color

Returns a light or dark color based on an input color's lightness:

```
contrast-color(input-color, dark-color, light-color, threshold)
```

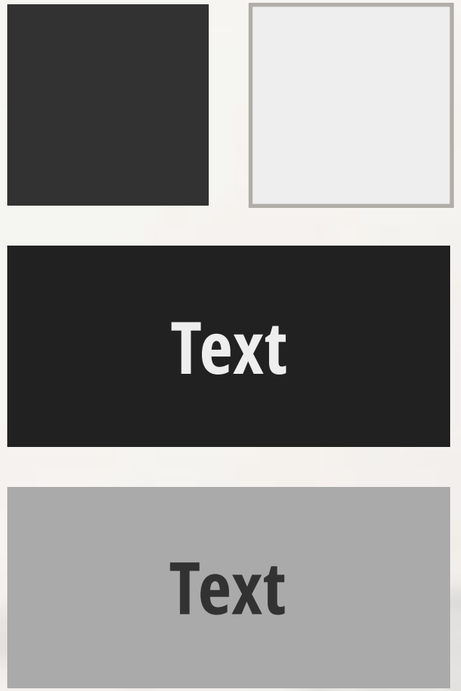

application.sass

```
@function button-text($bg)
  @return // which color?

.btn-a
  background: #222
  color: button-text(#222)

.btn-b
  background: #aaa
  color: button-text(#aaa)
```

3.3 Utilities & Stretch

application.sass

```
@function button-text($bg)
  @return contrast-color($bg, #333, #eee, 50%)

.btn-a
  background: #222
  color: button-text(#222) // #eee

.btn-b
  background: #aaa
  color: button-text(#aaa) // #333
```



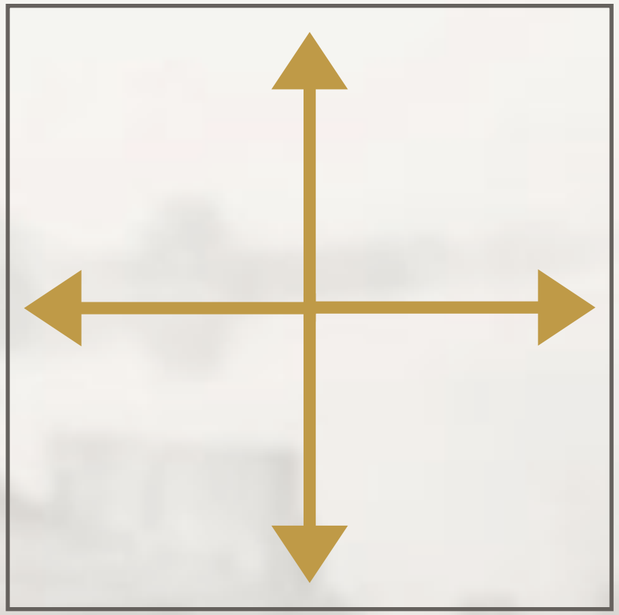
3.3 Utilities & Stretch

application.sass

```
.content
  height: 400px
  position: relative
  width: 400px

.sidebar
  background: #be9947
```

index.html

```
<section class="content">
  <aside class="sidebar"></aside>
</section>
```



we'd like .sidebar to stick to .content's edges

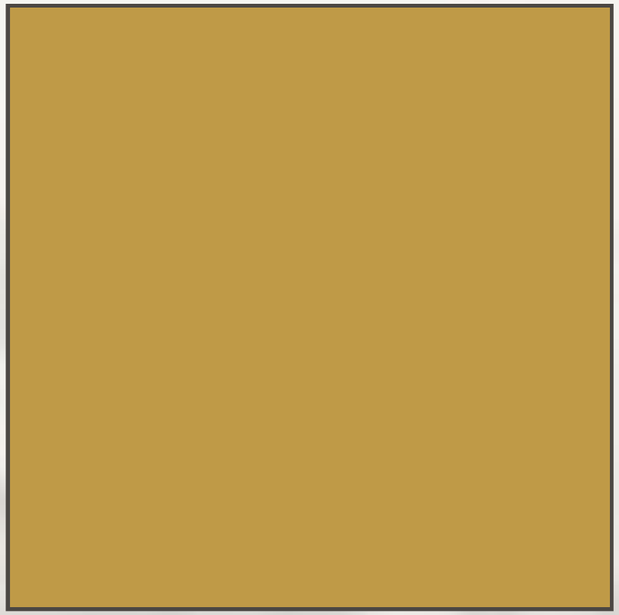
3.3 Utilities & Stretch

application.sass

```
.content
  height: 400px
  position: relative
  width: 400px

.sidebar
  background: #be9947
  position: absolute
  top: 0
  right: 0
  bottom: 0
  left: 0
```

index.html

```
<section class="content">
  <aside class="sidebar"></aside>
</section>
```



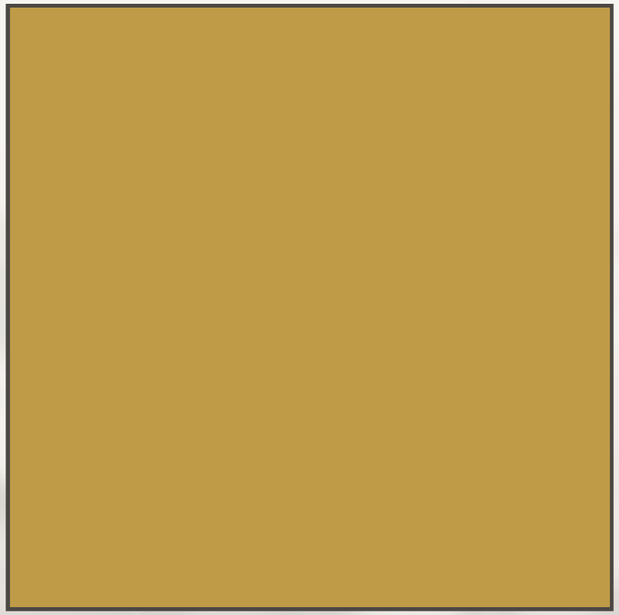
3.3 Utilities & Stretch

application.sass

```
@import "compass/layout"

.content
  height: 400px
  position: relative
  width: 400px

.sidebar
  background: #be9947
  +stretch
```

index.html

```
<section class="content">
  <aside class="sidebar"></aside>
</section>
```



3.3 Utilities & Stretch

Stretch

Outputs positioning for each side of a container:

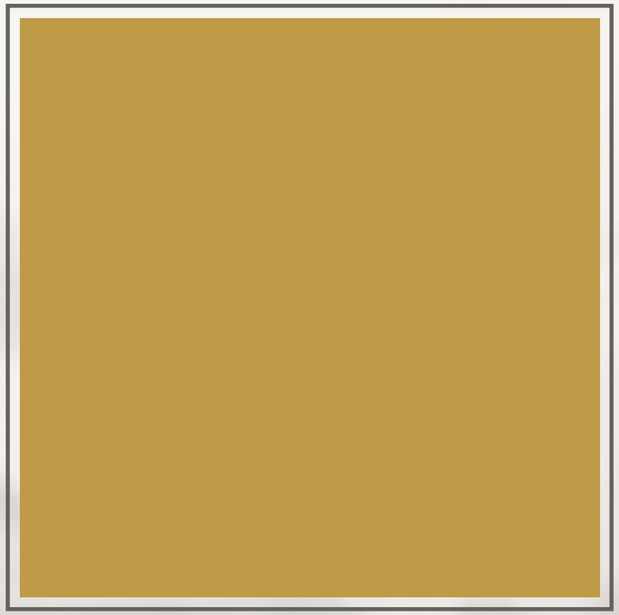
```
+stretch(0, 0, 0, 0)
```

application.sass

```
@include "compass/layout"

.content
  height: 400px
  position: relative
  width: 400px

.sidebar
  background: #be9947
  +stretch(5px, 5px, 5px, 5px)
```

index.html

```
<section class="content">
  <aside class="sidebar"></aside>
</section>
```



3.3 Utilities & Stretch

application.sass

```
@include "compass/layout"

.content
  height: 400px
  position: relative
  width: 400px

.sidebar
  background: #be9947
  +stretch(5px, auto, 5px, 5px)
  width: 100px
```

index.html

```
<section class="content">
  <aside class="sidebar"></aside>
</section>
```

3.3 Utilities & Stretch

s.png 236 x 236

application.sass

```
@import "compass"

.logo
  background: image-url('s.png')
  height: 236px
  width: 236px
```

application.css

```
.logo {
  background: url('/images/s.png');
  height: 236px;
  width: 236px;
}
```

3.4 Image Dimensions

application.sass

```
@import "compass"

.logo
  background: image-url('s.png')
  height: image-height('s.png')
  width: 236px
```

application.css

```
.logo {
  background: url('/images/s.png');
  height: 236px;
  width: 236px;
}
```

3.4 Image Dimensions

application.sass

```
@import "compass"

.logo
  background: image-url('s.png')
  height: image-height('s.png')
  width: image-width('s.png')
```

application.css

```
.logo {
  background: url('/images/s.png');
  height: 236px;
  width: 236px;
}
```

3.4 Image Dimensions

Use `inline-image` to base64 embed images into CSS:

application.sass

```
@import "compass"

.logo
  background: inline-image('s.png')
  height: image-height('s.png')
  width: image-width('s.png')
```

application.css

```
.logo {
  background: url('
EUgAAAGQAAABkCAMAAABHPGVmAAAAG
XRFWHRTb2Z0d2FyZQBBZO30bfS0uzz
2vD2rd3qkdLj4fP3ULjS2O/1...');
  height: 236px;
  width: 236px;
}
```

Inline Image

- Embeds the image into CSS (no more loading flash)
- ~10% image size increase, but reduces HTTP requests
- IE8+ (max 32KB), good mobile support